6. I/O Registers

The TMS34010 has 28 on-chip I/O registers that control and monitor the following functions:

- Host interface communications
- Local memory interface control
- Interrupt control
- Video timing and screen refresh

This section describes these functions, I/O register addressing, and then provides an alphabetical presentation of the I/O registers:

6.1 I/O Register Addressing

TMS34010 I/O registers occupy addresses >C000 0000 to >C000 01FF. These registers can be directly accessed by the GSP; they can also be indirectly accessed by a host processor through the host interface registers. For example, the host processor can indirectly read the contents of the PSIZE register by loading the address >C000 0150 into the HSTADRL and HSTADRH registers, and reading the HSTDATA register. Figure 6-1 illustrates the I/O register memory map.

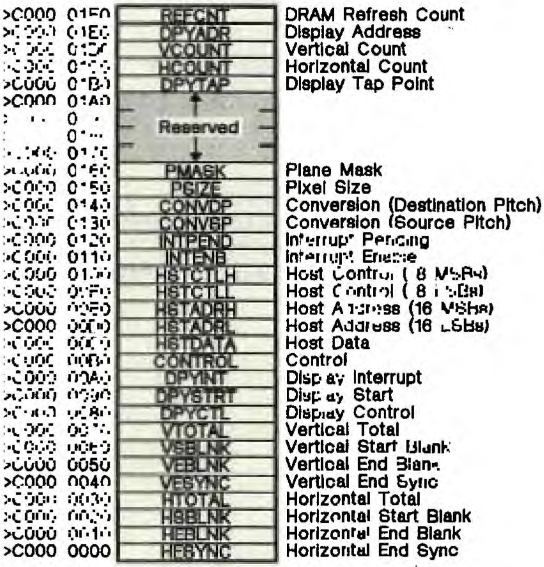


Figure 6-1. I/O Register Memory Map

The two MSBs of an I/O register's 32-bit internal address are not output on the TMS34010 pins; however, the address is fully decoded internally. Thus, the two MSBs of a 32-bit address must both be 1s for an address to be recognized as that of an I/O register. When an I/O register is accessed, the accompanying memory cycle (as seen at the TMS34010 pins) is altered so that the row address strobe is output, but the column address strobe is inhibited. This is true whether the access is initiated directly by the GSP or indirectly by a host processor.

An access of any address in the range >C000 0000 to >C000 01FF is decoded as an access of an on-chip register location, and the column address strobe remains inactive high through the cycle. An access of any location outside this range is treated as an access of an external memory location.

All I/O registers, with one exception, are cleared to 0 at reset. The exception is the HLT (halt) bit in the HSTCTL register, which is set depending on the value at the $\overline{HCS}$ input pin at the end of the reset pulse:

- If $\overline{HCS}$ is high at reset, the HLT bit is set to 1
- If $\overline{HCS}$ is low at reset, the HLT bit is set to 0

6.2 Latency of Writes to I/O Registers

When an instruction alters the contents of an I/O register, the memory write cycle that modifies the register may not be completed before execution of the next instruction begins. If the second instruction relies on the I/O register value loaded by the first instruction, the second instruction may cause incorrect results. This situation is easily avoided by ensuring that the write to the I/O register is allowed to complete before the I/O register value is used as an implied operand by a subsequent instruction. For example, by immediately following a write to an I/O register with a read of the register, the write is certain to have been completed by the time subsequent instructions begin execution.

Internal to the TMS34010, the memory controller operates semi-autonomously with respect to the execution unit that processes instructions. Parallelism between the execution unit and memory controller may allow a write initiated by an instruction to be completed only after one or more subsequent instructions have been executed. An instruction that alters an I/O register (or any other address in memory) transmits its request for a write cycle to the memory controller. Once the request is accepted, the memory controller is responsible for completing the write cycle; in the meantime, execution of the next instruction can begin.

A field insertion request submitted to the memory controller can take as many as five cycles to complete in the case in which a field of 18 or more bits straddles two word boundaries. This case requires a read-modify-write operation to one word, a write to a second word, and a read-modify-write operation to a third word. Although this would be an unusual way of altering locations in the I/O register file, it represents the theoretical worst case number of memory cycles for a field insertion. Other potential sources of delay to a pending field insertion request include:

- Screen-refresh cycle
- DRAM-refresh cycle
- Host-indirect read or write cycle
- Wait states required for slower memories
- Hold request from an external device

Any uncertainty as to whether a pending write to memory has been completed can be eliminated by making use of the fact that only one field insertion request can be queued at the memory controller at a time. An instruction that requests a second memory access before the earlier field insertion has been completed will be forced to wait. Hence, by following an instruction that alters an I/O register with an instruction that requests a second memory access (*any* memory access), the I/O register is certain to have been updated before the second instruction finishes executing.

6.3 I/O Registers Summary

Table 6-1 summarizes the I/O registers. Descriptions of the four categories of I/O registers follow the table.

Table 6-1. I/O Registers Summary

Host Interface Registers		
Register	**Address**	**Description**
HSTADRH	>C000 00E0	*Host interface address, high word.* Contains the 16 MSBs of a 32-bit pointer address used by a host processor for indirect accesses of TMS34010 local memory.
HSTADRL	>C000 00D0	*Host interface address, low word.* Contains the 16 LSBs of a 32-bit pointer address used by a host processor for indirect accesses of TMS34010 local memory.
HSTCTLH	>C000 0100	*Host interface control, high byte* Contains seven programmable bits that control host interface functions: NMI (bit 8) – Nonmaskable interrupt NMIM (bit 9) – NMI mode bit INCW (bit 11) – Increment pointer address on write INCR (bit 12) – Increment pointer address on read LBL (bit 13) – Lower byte last CF (bit 14) – Cache flush HLT (bit 15) – Halt TMS34010 execution Bits 0 through 7 and 10 are reserved
HSTCTLL	>C000 00F0	*Host interface control, low byte.* Contains eight programmable bits that control host interface functions: MSGIN (bits 0–2) – Input message buffer INTIN (bit 3) – Input interrupt bit MSGOUT (bits 4–6) – Output message buffer INTOUT (bit 7) – Output interrupt bit Bits 8 through 15 are reserved
Local Memory Interface Registers		
Register	**Address**	**Description**
CONTROL†	>C000 00B0	*Memory control.* Contains several parameters that control local memory interface operation: RM (bit 2) – DRAM refresh mode RR (bits 3–4) – DRAM refresh rate T (bit 5) – Transparency enable W (bits 6–7) – Window violation detection mode PBH (bit 8) – PixBlt horizontal direction PBV (bit 9) – PixBlt vertical direction PPOP (bits 10–14) – Pixel processing operation select CD (bit 15) – Cache disable Bits 0 and 1 are reserved
CONVDP†	>C000 0140	*Destination pitch conversion factor.* Used during XY to linear conversion of a destination memory address.
CONVSP†	>C000 0130	*Source pitch conversion factor.* Used during XY to linear conversion of a source memory address.

† Implied graphics operands

Table 6-1. I/O Registers Summary (Continued)

Register	Address	Description
Local Memory Interface Registers (Continued)		
PMASK†	>C000 0160	*Plane mask register.* Selectively enables/disables the various planes in the bit map of a display system in which each pixel is represented by multiple bits.
PSIZE†	>C000 0150	*Pixel size register.* Specifies the pixel size (in bits). Possible pixel sizes include 1, 2, 4, 8, and 16 bits.
REFCNT	>C000 01F0	*Refresh count register.* Generates the addresses output during DRAM refresh cycles and counts the intervals between successive DRAM refresh cycles: RINTVL (bits 2–7) – Specifies the refresh interval ROWADR (bits 8–15) – Row address Bits 0 and 1 are reserved
Interrupt Control Registers		
Register	**Address**	**Description**
INTENB	>C000 0110	*Interrupt enable.* Contains the interrupt mask used to selectively enable/disable the three internal and two external interrupts: X1E (bit 1) – External interrupt 1 enable X2E (bit 2) – External interrupt 2 enable HIE (bit 9) – Host interrupt enable DIE (bit 10) – Display interrupt enable WVE (bit 11) – Window violation interrupt enable Bits 0, 3 through 8, and 12 through 15 are reserved
INTPEND	>C000 0120	*Interrupt pending.* Indicates which interrupt requests are currently pending: X1P (bit 1) – External interrupt 1 pending X2P (bit 2) – External interrupt 2 pending HIP (bit 9) – Host interrupt pending DIP (bit 10) – Display interrupt pending WVP (bit 11) – Window violation interrupt pending Bits 0, 3 through 8, and 12 through 15 are reserved
Video Timing and Screen Refresh Registers		
Register	**Address**	**Description**
DPYADR	>C000 01E0	*Display address.* Counts the number of scan lines output between successive screen refresh cycles and contains the source of the row and column addresses output during a screen refresh cycle: LNCNT (bits 0–1) – Scan line counter SRFADR (bits 2–15) – Screen refresh address
DPYCTL	>C000 0080	*Display control.* Contains several parameters that control video timing signals: HSD (bit 0) – Horizontal sync direction DUDATE (bits 2–9) – Display address update ORG (bit 10) – Screen origin select SRT (bit 11) – Shift register transfer enable SRE (bit 12) – Screen refresh enable DXV (bit 13) – Disable external video NIL (bit 14) – Noninterlaced video enable ENV (bit 15) – Enable video Bit 1 is reserved.
DPYINT	>C000 00A0	*Display interrupt.* Specifies the next scan line that will cause a display interrupt request.

† Implied graphics operands

Table 6-1. I/O Registers Summary (Concluded)

Video Timing and Screen Refresh Registers (Continued)		
Register	**Address**	**Description**
DPYSTRT	>C000 0090	*Display start address.* Provides control of the automatic memory-to-shift-register cycles necessary to refresh a screen: LCSTRT (bits 0–1) – Specifies the number of scan lines to be displayed between screen refreshes SRSTRT (bits 2–15)– Starting screen-refresh address
DPYTAP	>C000 01B0	*Display tap point address.* Contains a VRAM tap point address output during shift register transfer cycles.
HCOUNT	>C000 01C0	*Horizontal count.* Counts the number of VCLK periods per horizontal scan line.
HEBLNK	>C000 0010	*Horizontal end blank.* Designates the endpoint for horizontal blanking.
HESYNC	>C000 0000	*Horizontal end sync.* Specifies the endpoint of the horizontal sync interval.
HSBLNK	>C000 0020	*Horizontal start blank.* Specifies the starting point of the horizontal blanking interval.
HTOTAL	>C000 0030	*Horizontal total.* Specifies the total number of VCLK periods per horizontal scan line.
VCOUNT	>C000 01D0	*Vertical count.* Counts the horizontal scan lines in a video display.
VEBLNK	>C000 0050	*Vertical end blank.* Specifies the endpoint of the vertical blanking interval.
VESYNC	>C000 0040	*Vertical end sync.* Specifies the endpoint of the vertical sync pulse.
VSBLNK	>C000 0060	*Vertical start blank.* Specifies the starting point of the vertical blanking interval.
VTOTAL	>C000 0070	*Vertical total.* Specifies the value of VCOUNT at which the vertical sync pulse begins.

6.3.1 Host Interface Registers

Five I/O registers are dedicated to host interface communications, allowing the TMS34010 to:

- Directly transfer status messages or command information
- Indirectly transfer large blocks of data through local memory
- Receive interrupt requests from a host processor
- Transfer interrupt requests to a host processor

The ability to indirectly transfer large blocks of data makes the host interface extremely flexible. For example, a host can transfer blocks of commands to the GSP, can halt the GSP temporarily to download a new program for the GSP to execute, or can read blocks of graphics data generated by the GSP.

The host interface registers occupy five GSP register locations, and are typically mapped into four consecutive 16-bit locations in the memory or I/O address space of the host processor. The host processor accesses the HSTCTLL and HSTCTLH registers as the eight LSBs and eight MSBs, respectively, of a single location (the HSTCTL register).

The HSTCTL (host control) register controls functions such as the transfer of interrupt requests and 3-bit status codes between a host processor and the TMS34010. These requests are typically used by software to coordinate the transfer of large blocks of data through GSP local memory. The HSTCTL register also allows the host to flush the instruction cache, halt GSP execution, and transmit nonmaskable interrupt requests to the GSP.

The host processor uses the remaining three host interface registers to indirectly access selected data blocks within GSP local memory. The HSTADRL and HSTADRH registers contain a 32-bit address that points to the current word location in memory. The HSTDATA register buffers data transferred to and from the memory under control of the host processor. The host interface can be programmed to automatically increment the address pointer following each transfer, providing the host with rapid access to a block of sequential locations.

6.3.2 Local Memory Interface Registers

Six of the I/O registers support local memory interface functions such as:

- Frequency of DRAM refresh cycles
- Type of DRAM refresh cycles
- Pixel size
- Color plane masking
- Various pixel access control parameters

6.3.3 Interrupt Interface Registers

Two I/O registers monitor and mask interrupt requests to the TMS34010. These include two external and three internal interrupts. External interrupt requests are transmitted to the GSP via input pins $\overline{\text{LINT}}1$ and $\overline{\text{LINT}}2$. The GSP can be programmed to generate an internal interrupt request in response to any of the following conditions:

- *Window violation* – an attempt is made to write a pixel to a location inside or outside a specified window, depending on the selected windowing mode.
- *Host interrupt* – the host processor sets the INTIN interrupt request bit in the HSTCTL register.
- *Display interrupt* – the specified line number in a frame is displayed on the monitor.

A nonmaskable interrupt occurs when the host processor sets the NMI bit in the HSTCTL host interface register. Reset is controlled by a dedicated pin.

6.3.4 Video Timing and Screen Refresh Registers

Fifteen I/O registers support video timing and screen refresh functions. The TMS34010's on-chip CRT timing generator creates the sync and blanking signals used to drive the CRT monitor in a bit-mapped display system. The timing of these signals can be controlled through the appropriate I/O registers, allowing the GSP to support various screen resolutions and interlaced or noninterlaced video.

The GSP directly supports VRAMs by generating the memory-to-shift-register cycles necessary to refresh the screen of a CRT monitor. Programmable features include the locations in memory to be displayed on the monitor, as well as the number of horizontal scan lines displayed between individual screen-refresh cycles.

The GSP can optionally be programmed to synchronize to externally generated sync signals. This permits GSP-created graphics images to be superimposed upon externally-created images. This external sync mode can also be used to synchronize the video timing of two or more GSP chips in a multiple-GSP display system.

6.4 Alphabetical Listing of I/O Registers

The remainder of this section describes the I/O registers individually; they are listed in alphabetical order. Fields within each register are identified and functions associated with each register are discussed.

Bits within I/O registers that are identified as *reserved* are not used by the TMS34010. When read, a reserved bit returns the last value written to it. No control function, however, is affected by this value. All reserved bits are loaded with 0s at reset. A good software practice is to maintain 0s in these bits.

Address >C000 00B0

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CD	PPOP					PBV	PBH	W		T	RR		RM	reserved	

Fields

Bits	Name	Function
0–1	Reserved	Not used
2	RM	DRAM refresh mode
3–4	RR	DRAM refresh rate
5	T	Pixel transparency enable
6–7	W	Window violation detection mode
8	PBH	PixBlt horizontal direction
9	PBV	PixBlt vertical direction
10–14	PPOP	Pixel processing operation select
15	CD	Instruction cache disable

Description The CONTROL register contains several control parameters used to configure local memory interface operation.

- **RM** (*DRAM refresh mode select*)

The RM bit selects the type of DRAM refresh cycle to be performed. Depending on the value of this bit, the GSP will perform each DRAM-refresh cycle as either a $\overline{\text{RAS}}$-only cycle or as a $\overline{\text{CAS}}$-before-$\overline{\text{RAS}}$ cycle. DRAMs and VRAMs that rely on the GSP to generate an 8-bit row address during a refresh cycle will typically use the $\overline{\text{RAS}}$-only refresh cycle, while those that generate their own 9-bit row address internally will use the $\overline{\text{CAS}}$-before-$\overline{\text{RAS}}$ refresh cycle.

RM	Description
0	Selects $\overline{\text{RAS}}$-only refresh cycle
1	Selects $\overline{\text{CAS}}$-before-$\overline{\text{RAS}}$ refresh cycle

- **RR** (*DRAM refresh rate*)

The RR field controls the frequency of DRAM refresh cycles. The GSP automatically generates DRAM refresh cycles at regular intervals. The duration of the interval is specified by the value of RR. If required, DRAM refreshing can be disabled by setting RR to the appropriate value.

The initial value of RR after reset is 00. No DRAM refresh cycles are performed while the GSP $\overline{\text{RESET}}$ signal is active.

RR	Description
00	Refresh every 32 local clock periods
01	Refresh every 64 local clock periods
10	Reserved code
11	No DRAM refreshing

- **T** (*Pixel transparency enable*)

  The T bit enables or disables the pixel attribute of transparency. When transparency is enabled, a value of 0 resulting from a pixel operation on source and destination pixels is inhibited from overwriting the destination pixel. In the case of a replace operation, a source pixel value of 0 is inhibited from overwriting the destination pixel. Disabling transparency allows a pixel value of 0 to be written to the destination.

T	Effect
0	Disable transparency
1	Enable transparency

- **W** (*Window violation detection mode*)

  The W field selects the course of action to be taken when a pixel operation will cause a pixel to be written to a location lying either inside or outside the specified window limits. Window checking applies only to attempts to write to pixel locations defined by XY addresses; writes to pixel locations defined by linear memory addresses are not affected. Nonpixel data writes are not affected.

W	Description
00	No pixel writes are inhibited, and no interrupt requests are generated
01	Generate interrupt request on attempt to write to pixel lying inside window, and inhibit all pixel writes
10	Generate interrupt request on attempt to write to pixel lying outside window
11	Inhibit pixel writes outside window, but do not request interrupt

  A request for a window violation interrupt can occur when W=01 or W=10. The WVP bit in the INTPEND register is set to 1 to indicate that a window violation has occurred. This in turn causes the GSP to be interrupted if the WVE bit in the INTENB register and the status IE bit are set to 1.

- **PBH** (*PixBlt horizontal direction control*)

  The PBH bit determines the horizontal direction (increasing X or decreasing X) of pixel processing for the following instructions:

  - PIXBLT XY,XY
  - PIXBLT L,XY
  - PIXBLT XY,L
  - PIXBLT L,L

PBH	Effect
0	Increment X (move from left to right)
1	Decrement X (move from right to left)

- **PBV** (*PixBlt vertical direction control*)

  The PBV bit determines the vertical direction (increasing Y or decreasing Y) of pixel processing for the following instructions:

  - PIXBLT XY,XY
  - PIXBLT L,XY
  - PIXBLT XY,L
  - PIXBLT L,L

PBV	Effect†
0	Increment Y (move from top to bottom)
1	Decrement Y (move from bottom to top)

† Default screen origin assumed

- **PPOP** (*Pixel processing operation select*)

  The PPOP field selects the operation to be performed on the source and destination pixels during a pixel operation. The following 16 PPOP codes perform Boolean operations on pixels of 1, 2, 4, 8, and 16 bits.

PPOP	Operation	Description
00000	S → D	Replace destination with source
00001	S AND D → D	AND source with destination
00010	S AND $\overline{D}$ → D	AND source with NOT destination
00011	0 → D	Replace destination with 0s
00100	S OR $\overline{D}$ → D	OR source with NOT destination
00101	S XNOR D → D	XNOR source with destination
00110	$\overline{D}$ → D	Negate destination
00111	S NOR D → D	NOR source with destination
01000	S OR D → D	OR source with destination
01001	D → D	No change in destination†
01010	S XOR D → D	XOR source with destination
01011	$\overline{S}$ AND D → D	AND NOT source with destination
01100	1 → D	Replace destination with 1s
01101	$\overline{S}$ OR D → D	OR NOT source with destination
01110	S NAND D → D	NAND source with destination
01111	$\overline{S}$ → D	Replace destination with NOT source

† Although the destination array is not changed by this operation, memory cycles still occur.

The following six PPOP codes perform arithmetic operations on 4-, 8-, and 16-bit pixels (but not 1 or 2 bits).

PPOP	Operation	Description
10000	D + S → D	Add source to destination
10001	ADDS(D,S) → D	Add S to D with saturation
10010	D - S → D	Subtract source from destination
10011	SUBS(D,S) → D	Subtract S from D with saturation
10100	MAX(D,S) → D	Maximum of source and destination
10101	MIN(D,S) → D	Minimum of source and destination

PPOP codes 10110 through 11111 are reserved.

Standard addition and subtraction allow the result of the operation to overflow. However, add-with-saturation and subtract-with-saturation (ADDS and SUBS) do not allow overflow or underflow. In cases in which addition would allow an overflow, ADDS produces a result whose value is all 1s. In cases in which subtraction would allow an underflow, SUBS produces a result whose value is all 0s.

- **CD** (*Cache disable*)

The CD bit selectively enables or disables the instruction cache.

CD	Effect
0	Enable instruction cache
1	Disable instruction cache

When the cache is disabled, cache contents (including data, P flags, SSA registers, and so on) remain undisturbed. While the cache remains disabled, all instructions are fetched from memory rather than cache. When the cache is subsequently enabled, its previous state (before it was disabled) is restored. The instructions retained within the cache are once again available for execution.

Address >C000 0140

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CONVDP															

Description CONVDP is a full 16-bit register that contains a control parameter used during execution of a pixel operation instruction. CONVDP is used with:

- XY addressing
- Window clipping
- PIXBLTs or FILLS (except for PIXBLT L,L) that process pixels from the bottom of the array to the top (PBV=1)

CONVDP is calculated as the result of an LMO instruction whose input operand is the destination pitch value in register B3 (DPTCH). The following GSP assembly code calculates the CONVDP value.

```
LMO  B3,B0           ; Convert DPTCH value
MOVE B0,@CONVDP,0    ; Place result in CONVDP register
```

In this example, B0 is used as a scratch register. Constant CONVDP has the value >C000 0140, and the size of Field 0 is 16 bits.

GSP internal hardware uses the CONVDP value during XY-to-linear conversion of a destination address. CONVDP is also used for corner adjust operations in the Y direction (when PBV=1). The value contained in the five LSBs of CONVDP should be the 1's complement of $\log_2$(DPTCH). When an XY address is specified for the destination, DPTCH must be a power of two; thus, $\log_2$(DPTCH) is an integer. During XY-to-linear conversion, the product of the Y value and the destination pitch is calculated by shifting Y left by $\log_2$(DPTCH).

One instruction, the PIXBLT XY,L instruction, specifies the destination address in linear format but also requires DPTCH to be a power of two. This restriction is necessary when the PBV bit is set to 1.

Address >C000 0130

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CONVSP															

Description CONVSP is a full 16-bit register that contains a control parameter used during execution of a pixel operation instruction. CONVSP is used with:

- XY addressing
- Window clipping
- PIXBLTs or FILLS (except for PIXBLT L,L) that process pixels from the bottom of the array to the top (PBV=1)

CONVSP is calculated as the result of an LMO instruction whose input operand is the source pitch value in register B1 (SPTCH). The following GSP assembly code calculates the CONVSP value

```
LMO  B1,B0          ; Convert SPTCH value
MOVE B0,@CONVSP     ; Place result in CONVSP register
```

In this example, B0 is used as a scratch register. Constant CONVSP has the value >C000 0130, and the size of Field 0 is 16 bits.

GSP internal hardware uses the CONVSP value during XY-to-linear conversion of a source address. CONVSP is also used for corner ajust operations in the Y direction (when PBV=1). The value contained in the five LSBs of CONVSP should be the 1's complement of $\log_2$(SPTCH). When an XY address is specified for the source, SPTCH must be a power of two; thus, $\log_2$(SPTCH) is an integer. During XY-to-linear conversion, the product of the Y value and the source pitch is calculated by shifting Y left by $\log_2$(SPTCH).

Two instructions that specify the source address in linear format also require SPTCH to be a power of two. This is necessary when window preclipping is required during execution of either of the following instructions:

- PIXBLT B,XY
- PIXBLT L,XY

It is also necessary when either of these two instructions is executed and the PBV bit in the CONTROL register is set to 1. If PBV=0 and window clipping is disabled, or if window clipping is enabled but the specified array does not require preclipping in the Y dimension, CONVSP is not used, and SPTCH is not required to be a power of two.

Address >C000 01E0

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRFADR														LNCNT	

Fields

Bits	Name	Function
0–1	LNCNT	Scan line counter
2–15	SRFADR	Screen refresh address

Description The 16-bit DPYADR register contains two separate counters that control the generation of screen-refresh cycles. A screen-refresh cycle transfers the video data for a new scan line to the shift registers of the VRAMs.

- **LNCNT** (*Scan line counter*)

  LNCNT counts the number of scan lines output to the screen between successive screen-refresh cycles. Providing explicit control over the line count permits the implementation of systems that do not reload the VRAMs' internal shift register on every horizontal scan line. The two-bit LNCNT field is loaded from the two-bit LCSTRT field of the DPYSTRT register at the end of each screen-refresh cycle. The value loaded determines whether the next screen-refresh cycle occurs after 1, 2, 3 or 4 scan lines:

  - When LCSTRT = 0, a screen refresh occurs after every line.
  - When LCSTRT = 1, 2 or 3, a screen-refresh cycle occurs after every 2, 3 or 4 lines, respectively.

- **SRFADR** (*Screen refresh address*)

  SRFADR is the source of the row and column addresses output during a screen-refresh cycle. The 14 bits of SRFADR are output as logical address bits 10–23 during screen-refresh cycles. During row address time, DPYADR4–DPYADR15 are output on LAD12–LAD23, and 0s are output on the remaining LAD pins (except as modified by the contents of the DPYTAP register). During column address time, DPYADR2–DPYADR6 are output on LAD6–LAD10 and 0s are output on the remaining LAD lines. Following the completion of each screen-refresh cycle, the value in SRFADR is decremented by the amount indicated in the DUDATE field of the DPYCTL register.

  The following diagrams illustrate the mapping of bits to LAD0–LAD15 from

  1) The logical address as seen by the programmer

     and

  2) The bits of the DPYADR register

The bits of a 32-bit logical address are numbered 0 to 31, beginning with the LSB. The 14 MSBs of DPYADR, shown in the diagram below, are output as logical address bits 10–23 during a screen-refresh cycle. DPYADR2 corresponds to logical address bit 10, DPYADR3 corresponds to logical address bit 11, and so on.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DPYADR	SRFADR														LNCNT	
Logical Address	23	22	21	20	19	18	17	16	15	14	13	12	11	10		

The next diagram shows the mapping of logical addresses to LAD0–LAD15 during the row and column address times of the cycle. The symbol *xx* indicates status information output with the row and column addresses.

LAD Pin No.:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Logical Row Address Bits:	XX	26	25	24	23	22	21	10	19	18	17	16	15	14	13	12	Row Address Time
Corresponding DPYADR Bits:					15	14	13	12	11	10	9	8	7	6	5	4	
Logical Column Address Bits:	XX	XX	29	28	27	14	13	12	11	10	9	8	7	6	5	4	Column Address Time
Corresponding DPYADR Bits:					7	6	5	4	3	2							

A board designer must select eight consecutive address lines from LAD0–LAD11 to connect to the multiplexed address inputs of the VRAMs. For example, by selecting the eight lines LAD2–LAD9, bits 14–21 of the logical address become the row address bits output to the RAMs, and bits 6–13 of the logical address become the column address bits. This means that during a screen-refresh cycle, bits 6–13 of DPYADR become the row address bits output to the RAMs, and bits 4–5 of DPYADR become the two MSBs of the tap point address.

Address >C000 0080

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENV	NIL	DXV	SRE	SRT	ORG	DUDATE								Res	HSD

Fields

Bits	Name	Function
0	HSD	Horizontal sync direction
1	Reserved	Not used
2–9	DUDATE	Display address update
10	ORG	Screen origin select
11	SRT	Shift register transfer enable
12	SRE	Screen refresh enable
13	DXV	Disable external video
14	NIL	Noninterlaced video enable
15	ENV	Enable video

Description The DPYCTL register contains several parameters that control video timing signals and shift-register transfer cycles using VRAMs.

- **HSD** (*Horizontal sync direction*)

The HSD bit controls the direction (input or output) of the $\overline{\text{HSYNC}}$ (horizontal sync) pin when the GSP is in external video mode (DXV=0). If HSD=0, $\overline{\text{HSYNC}}$ is configured as an input, the same as $\overline{\text{VSYNC}}$. In this case, the on-chip horizontal sync interval begins when either:

– The start of the external horizontal sync pulse input at the $\overline{\text{HSYNC}}$ pin is detected,

or

– HCOUNT = HTOTAL,

whichever condition occurs first.

$\overline{\text{VSYNC}}$ and $\overline{\text{HSYNC}}$ are configured as inputs or outputs according to the values of the HSD and DXV bits:

HSD	DXV	$\overline{\text{HSYNC}}$	$\overline{\text{VSYNC}}$
0	0	Input	Input
0	1	Output	Output
1	0	Output	Input
1	1	Undefined	

When $\overline{VSYNC}$ and $\overline{HSYNC}$ are both configured as inputs, the on-chip vertical sync interval begins when any of the following conditions occur:

- The start of the external vertical sync pulse input at the $\overline{VSYNC}$ pin is detected,

or

- VCOUNT=VTOTAL, and the start of the horizontal sync pulse input at the $\overline{HSYNC}$ pin is detected,

or

- VCOUNT=VTOTAL and HCOUNT=HTOTAL.

When $\overline{VSYNC}$ is an input and $\overline{HSYNC}$ is an output, the vertical sync interval begins when either the first or third of the listed conditions occurs.

- **DUDATE** (*Display update amount*)

The DUDATE field indicates the amount by which the SRFADR field in the DPYADR register is incremented (if ORG=0) or decremented (ORG=1) following completion of each memory-to-shift-register cycle used to refresh the screen. DUDATE is loaded with a value containing seven 0s and a single 1. The 1 indicates the bit position at which DPYADR is to be incremented (or decremented if ORG=1).

DUDATE	Increment Size
00000000	0
00000001	1
00000010	2
00000100	4
00001000	8
00010000	16
00100000	32
01000000	64
10000000	128

The increment size is undefined when more than one bit in the DUDATE field is a 1. When interlaced scan mode is enabled, SRFADR is incremented/decremented by half the value indicated in DUDATE at the start of a vertical blanking interval preceding the start of an odd field, just after DPYADR2–DPYADR15 have been loaded from DPYSTRT2–DPYSTRT15.

For noninterlaced scanning, DUDATE is programmed to increment the screen address by one scan line. For interlaced scanning, DUDATE is programmed to increment the screen address by two scan lines. Larger increments are typically not used since screen-refresh cycles do not occur more often than once per active scan line. In special applications, however, the value of DUDATE can be adjusted to achieve video effects such as vertical zoom in and zoom out. (Horizontal zoom must be implemented in the external shift register logic).

- **ORG** (*Screen origin select*)

The ORG bit controls the origin of the screen coordinate system.

ORG	Effect
0	XY coordinate origin located in upper left corner of screen
1	XY coordinate origin located in lower left corner of screen

If ORG=0 then DPYADR is updated by being incremented by the value in the DUDATE field. If ORG=1 then DPYADR is updated by being decremented by the value in the DUDATE field. Unless explicitly stated otherwise, the discussion in this document assumes that the default origin (ORG=0) is used.

- **SRT** (*Shift-register-transfer enable*)

The SRT bit enables conversion of an ordinary pixel access into a shift-register-transfer cycle.

SRT	Effect
0	Pixel access cycles occur normally
1	Pixel access cycles are converted into VRAM shift-register-transfer cycles

The TMS34010 instruction set includes several instructions (DRAV, PIXT, LINE, FILL, and PIXBLT) that operate specifically on pixels. By default, SRT=0 and memory accesses performed during accesses of pixel data are the usual memory read and write cycles. When SRT=1, however, accesses of pixel data are converted to shift-register-transfer cycles:

- A pixel read cycle is converted to a memory-to-shift-register cycle
- A pixel write cycle is converted to a shift-register-to-memory cycle

This shift-register-transfer cycle is performed under explicit program control, as opposed to the screen-refresh cycles enabled by the SRE bit, which are automatically generated at regular intervals.

Uses of the SRT bit include bulk initialization of the entire VRAM array: the entire screen can be cleared to a specified background color in only 256 memory cycles. (All VRAMs do not support this capability.) Only pixel accesses are affected by the state of the SRT bit. Instruction fetches and non-pixel data accesses are not altered in any way.

- **SRE** (*Screen-refresh enable*)

  The SRE bit enables automatic screen refreshing. Screen refreshes are performed by means of the VRAM memory-to-shift-register cycles which the GSP performs automatically during selected horizontal blanking intervals. The frequency of screen-refresh cycles and the generation of the addresses output during these cycles are programmed by means of the DPYSTRT and DPYCTL registers.

SRE	Effect
0	Disable screen refresh
1	Enable screen refresh

- **DXV** (*Disable external video*)

  The DXV bit selects between internally generated or externally generated video timing.

DXV	Effect
0	Selects external video source
1	Selects internally generated video timing

When DXV=0, the GSP video timing circuitry is programmed to lock onto an external video source. The $\overline{\text{VSYNC}}$ pin is configured as an input and is connected to an external vertical sync signal. If HSD=0, $\overline{\text{HSYNC}}$ is also configured as an input and is connected to an external horizontal sync signal.

When DXV=1, the GSP generates its own video timing, according to the values loaded into the video timing registers. The $\overline{\text{HSYNC}}$ and $\overline{\text{VSYNC}}$ pins are configured as outputs, and provide the horizontal and vertical sync signals required to drive the video monitor.

- **NIL** (*Noninterlaced video enable*)

  The NIL bit selects between an interlaced or a noninterlaced display. The video timing signals output by the GSP are modified according to this selection. The timing differences between interlaced and noninterlaced displays are described in Section 9.

NIL	Effect
0	Selects interlaced video timing
1	Selects noninterlaced video timing

- **ENV** (*Enable video*)

The ENV bit enables or disables the video display. The display remains blanked when ENV=0. During this time, the signal output at the $\overline{\text{BLANK}}$ pin is forced to remain at its active-low level throughout the frame, and setting of the DIP (display interrupt) bit in the INTPEND register is inhibited. (If DIP is already set at the time the ENV is changed from 1 to 0, DIP remains set until explicitly cleared.) When ENV=1, the video display is enabled. The $\overline{\text{BLANK}}$ output signal is controlled according to the parameters contained in the video timing registers, and the DIP bit becomes set when the condition VCOUNT = DPYINT occurs.

ENV	Effect
0	Blank entire screen
1	Enable video

Address >C000 00A0

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DPYINT															

Description

The DPYINT register designates the next scan line at which a display interrupt will be requested. This register facilitates the coordination of software activity with the refreshing of selected horizontal lines on the screen of a video monitor.

The contents of DPYINT are compared to the VCOUNT register. When VCOUNT = DPYINT, a display interrupt is requested and the DIP bit in the INTPEND register is set to 1. This coincides with the start of the horizontal blanking interval that marks the end of the line designated by the value contained in DPYINT.

For split-screen applications, a new value can be loaded into the DPYADR register immediately following detection of the 0-to-1 transition of DIP. The new DPYADR value will not affect the line that immediately follows the end of the current horizontal blanking interval, but will affect the next line. The details of this timing are as follows. A screen-refresh cycle may be scheduled to occur at the start of the same horizontal blanking interval during which DIP becomes set. At the end of the screen-refresh cycle, the screen-refresh address in the DPYADR register will be automatically incremented. Requests for screen-refresh cycles have a higher priority than requests for cycles initiated by the on-chip processor. Hence, if the processor loads a new value into DPYADR immediately following detection of DIP's transition from 0 to 1, the value will become the address used for the next screen-refresh cycle, which cannot occur before the next horizontal blanking interval. Between the time that DIP becomes set to 1 and the completion of the next screen-refresh cycle at least one full scan line later, the DPYADR register is guaranteed not to be incremented. Its contents will change during this interval only if it is loaded with a new value under explicit program control. The display interrupt is disabled when the ENV bit in the DPYCTL register is 0.

Address >C000 0090

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRSTRT														LCSTRT	

Fields

Bits	Name	Function
0–1	LCSTRT	Starting line count
2–15	SRSTRT	Starting screen-refresh address

Description The DPYSTRT register contains two parameters that control the automatic memory-to-shift-register cycles necessary to refresh the screen.

- **LCSTRT** (*Starting line count*)

  LCSTRT is a two-bit code designating the number of scan lines to be displayed between screen refreshes.

LCSTRT Value	Scan Lines Between Refresh Cycles
0 0	1
0 1	2
1 0	3
1 1	4

  LCSTRT is loaded into the LNCNT field of the DPYADR register at the end of each screen-refresh cycle. LCSTRT is also loaded into LNCNT at the start of the last horizontal blanking interval preceding the first active scan line of a new frame.

- **SRSTRT** (*Starting screen-refresh address*)

  The 14-bit SRSTRT field contains the starting address loaded into the DPYADR register at the start of each frame. Its value identifies the start of the region of the graphics bit map to be displayed on the screen. SRSTRT is loaded into the SRFADR field of the DPYADR register at the beginning of each vertical blanking interval. (Loading occurs coincides with the start of the horizontal blanking interval at the end of the last active scan line in the frame.)

  The sense of the SRSTRT value depends on the value of the ORG (origin select) bit in the DPYCTL register. When ORG=0, SRSTRT is loaded with the **1's complement** of the starting address. When ORG=1, SRSTRT is loaded with the unmodified starting address. Regardless of the value of the ORG bit, the starting address points to the location in memory of the first pixel output to the screen during each frame. For a typical CRT display, the first pixel of each frame is output to the top left corner of the screen. Refer to the description of the DPYADR register for more information on the generation of screen-refresh addresses.

DPYTAP Display Tap Point Address Register DPYTAP

Address >C000 01B0

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		DPYTAP													

Fields

Bits	Name	Function
0–13	DPYTAP	Display tap point address
14–15	Reserved	Not used

Description

The DPYTAP register contains a VRAM tap point address output during a screen-refresh (memory-to-shift-register) cycle. (The contents of DPYTAP are not output during a shift-register transfer initiated under program control while the SRT bit in the DPYCTL register is set to 1.) During a screen-refresh cycle, the 16 bits of the DPYTAP register are logical ORed with the value output at the LAD0–LAD15 pins during the column address time. DPYTAP bit 0 is ORed with LAD0, DPYTAP bit 1 is ORed with LAD1, and so on. This means that the column address output during the cycle is the OR of bits 2–7 of DPYADR and bits 0–15 of DPYTAP.

One application of the DPYTAP register is to permit horizontal panning of the screen over a frame buffer that is wider than the screen. A DPYTAP value of 0 locates the screen at its leftmost position within the frame buffer. Incrementing DPYTAP causes the display to pan to the right through the frame buffer.

DPYTAP is typically used to alter (set to a value other than all 0s) only those column address bits of the SRFADR field of DPYADR that are never incremented. For instance, given a VRAM that requires an 8-bit column address, assume that SRFADR alternately sets the two MSBs of the column address to 00, 01, 10, and 11. In this case, DPYTAP should contain 1s only in the bit positions corresponding to the six LSBs of the column address.

Address >C000 01C0

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HCOUNT															

Description

The HCOUNT register is a 16-bit counter used in the generation of the horizontal sync and blanking signals. HCOUNT is incremented on the falling edge of the video input clock, and is used to count the number of video clock periods per horizontal scan line. To generate horizontal sync and blanking signals, the value of HCOUNT is compared to the value of the four horizontal timing registers: HESYNC, HEBLNK, HSBLNK, and HTOTAL. When external sync mode is disabled and the value in HCOUNT = HTOTAL, HCOUNT is reset to 0 on the next VCLK falling edge and the $\overline{\text{HSYNC}}$ output is driven active low. HCOUNT is also reset to 0 if the external sync mode is enabled and the input signal $\overline{\text{HSYNC}}$ is driven low.

Two separate, asynchronous elements of the GSP logic can access the HCOUNT register:

- The internal processor, which runs synchronously to local clocks LCLK1 and LCLK2, can access HCOUNT as an I/O register.
- The video timing control logic, which runs synchronously to the video clock VCLK, increments and clears HCOUNT in generating the sync and blanking signals.

No synchronization between these two subsystems is provided, and HCOUNT can only be reliably read or written to while VCLK is held at the logic-high level. HCOUNT is typically not read or written to except during chip test.

Address >C000 0010

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HEBLNK															

Description The HEBLNK register is used during the generation of the blanking signal output to the video monitor. The 16-bit value loaded into HEBLNK is compared to HCOUNT, and designates the point at which the horizontal blanking interval ends. The blanking signal output at the $\overline{\text{BLANK}}$ pin is a composite of the internal horizontal and vertical blanking signals. When the value in HCOUNT = HEBLNK, the $\overline{\text{BLANK}}$ output is driven inactive high unless vertical blanking is currently active. Most video monitors require HEBLNK to be set to a value that is less than the value in HSBLNK, but greater than the value in HESYNC.

HESYNC Horizontal End Sync Register HESYNC

Address >C000 0000

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HESYNC															

Description

The HESYNC register is used during generation of the horizontal sync signal output to the video monitor. The 16-bit value loaded into HESYNC determines the point at which the horizontal sync pulse ends. When the value in HCOUNT = HESYNC, the signal output from the $\overline{\text{HSYNC}}$ pin is driven inactive high to signal the end of the horizontal sync interval. Typical monitors require that HESYNC be set to a value less than the value contained in the HEBLNK register. (However, the HESYNC value is not *required* to be less than the HEBLNK value.) The minimum value of HESYNC is 0.

When external video is enabled and the $\overline{\text{HSYNC}}$ pin is configured as an input, HESYNC should be loaded with a value that ensures that the condition HCOUNT = HESYNC occurs after the external $\overline{\text{HSYNC}}$ signal has gone inactive-high, but before $\overline{\text{HSYNC}}$ goes active low again. For example, a good HESYNC value might be the average of the values in HEBLNK and HSBLNK.

Address >C000 0020

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HSBLNK															

Description

The HSBLNK register is used during generation of the blanking signal output to the video monitor. The 16-bit value in HSBLNK is compared to HCOUNT, and designates the point at which the horizontal blanking interval begins. The blanking signal output at the $\overline{\text{BLANK}}$ pin is a composite of the internal horizontal and vertical blanking signals. When the condition HCOUNT = HSBLNK occurs, the $\overline{\text{BLANK}}$ output is driven from its inactive-high level to its active-low level (unless it is already low due to vertical blanking being active).

Several internal events coincide with the start of horizontal blanking. First, when a screen-refresh cycle is programmed to occur during a particular horizontal scan line, a request for the cycle is sent to the memory controller at the beginning of the horizontal blanking interval that occurs at the end of the line. Second, if a display interrupt request is programmed to occur during a particular horizontal scan line, the request is generated at the start of horizontal blanking. Typical monitors require that HSBLNK be set to a value that is less than the value in HTOTAL, but greater than the value in HEBLNK.

Host Interface Register, High Word

Address >C000 00E0

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HSTADRH															

Description

The HSTADRH register contains the 16 MSBs of a 32-bit pointer address; the 16 LSBs are contained in HSTADRL. The contents of HSTADRL and HSTADRH are concatenated to form a single 32-bit address during an indirect access by a host processor. The pointer address can be accessed by both the host processor and the GSP. The host accesses the pointer address through two 16-bit host interface registers that are mapped into the host's memory or I/O address space.

The four LSBs of the 32-bit pointer address are forced to 0 to point to an even word boundary in memory. If the address pointer is incremented past the largest word address in memory, it will wrap around to the lowest address (all 0s).

When you use the HSTADRH and HSTADRL registers to read data indirectly from the host, be sure that you access them in the correct order. If LBL=0, HSTADRH should be written last. If LBL=1, HSTADRL should be written last.

> **Note:**
>
> When the TMS34010's on-chip processor writes to HSTADRH or HSTADRL, the referenced data is **not** automatically read into HSTDATA. The host must perform one of two operations to read the referenced data:
>
> 1) If INCR=0, the host processor reads the HSTDATA register twice. The second read provides valid data.
>
> 2) If INCR=1 or is unknown, the host processor reads and then writes the HSTADRH register (if LBL=0), or the HSTADRL register (if LBL=1). The HSTDATA register then contains valid data. If LBL is unknown, both HSTADRH and HSTADRL may be read and then written to make HSTDATA valid.

Host Interface Register, Low Word

Address >C000 00D0

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HSTADRL															

Description

The HSTADRL register contains the 16 LSBs of a 32-bit pointer address; the 16 MSBs are contained in HSTADRH. The contents of HSTADRL and HSTADRH are concatenated to form a single 32-bit address during an indirect access by a host processor. The pointer address can be accessed by both the host processor and the GSP. The host accesses the pointer address through two 16-bit host interface registers that are mapped into the host's memory or I/O address space.

The four LSBs of the 32-bit pointer address are forced to 0 to point to an even word boundary in memory. If the address pointer is incremented past the largest word address in memory, it will wrap around to the lowest address (all 0s).

When you use the HSTADRH and HSTADRL registers to read data indirectly from the host, be sure that you access them in the correct order. If LBL=0, HSTADRH should be written last. If LBL=1, HSTADRL should be written last.

Note:

When the TMS34010's on-chip processor writes to HSTADRH or HSTADRL, the referenced data is **not** automatically read into HSTDATA. The host must perform one of two operations to read the referenced data:

1) If INCR=0, the host processor reads the HSTDATA register twice. The second read provides valid data.

2) If INCR=1 or is unknown, the host processor reads and then writes the HSTADRH register (if LBL=0), or the HSTADRL register (if LBL=1). The HSTDATA register then contains valid data. If LBL is unknown, both HSTADRH and HSTADRL may be read and then written to make HSTDATA valid.

Host Interface Control Register, High Byte

HSTCTLH

Address >C000 0100

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HLT	CF	LBL	INCW	INCR		NMIM	NMI								

Fields

Bits	Name	Function
0–7	Reserved	Not used
8	NMI	Nonmaskable interrupt
9	NMIM	Mode bit for NMI
10	Reserved	Not used
11	INCW	Increment pointer address on write
12	INCR	Increment pointer address on read
13	LBL	Lower byte last
14	CF	Cache flush
15	HLT	Halt GSP processing

Description

The HSTCTLH register contains seven programmable bits used to control host interface communications. A host processor can access the control bits in the HSTCTLL and HSTCTLH registers as a single host interface register, HSTCTL. The bits of the host interface's HSTCTL register are mapped into two separate I/O register locations in the GSP's memory map, HSTCTLL and HSTCTLH, to allow the GSP to alter the bits in one location without affecting the bits in the other.

The HSTCTLH bits can be both written to and read by both the host processor and the GSP. Unpredictable results will occur if the GSP and host simultaneously write different values to the HSTCTLH bits. Typically only the host alters the bits in HSTCTLH.

- **NMI** (*Nonmaskable interrupt, host to GSP*)

  The nonmaskable interrupt allows the host processor to redirect the execution flow of GSP processing to an NMI routine, regardless of the current state of the interrupt mask flags. The host writes a 1 to the NMI bit to send a nonmaskable interrupt request to the GSP. The interrupt request cannot be disabled, and will always be executed (unless the GSP is reset before it can complete interrupt execution). The interrupt is initiated immediately upon NMI becoming set (at the time the current instruction completes execution, or in the case of a pixel array instruction, at the next interruptible point in the instruction). Once the interrupt is taken, internal logic automatically clears the NMI bit to 0.

  One use of the NMI is to generate a soft reset after the host downloads new program code into GSP memory. Following execution of a nonmaskable interrupt, screen-refresh and DRAM-refresh functions continue unaffected. The contents of internal registers other than the HSTCTL register are not altered by the interrupt, although they can be modified by the NMI service routine.

- **NMIM** (*Nonmaskable interrupt mode*)

  The NMI mode bit determines whether or not the context of the interrupted program is saved when a nonmaskable interrupt occurs. When NMIM=0, the context is saved on the system stack before the NMI service routine is executed. When NMIM=1, the context is discarded when the NMI service routine is executed.

  The NMIM=0 mode supports applications such as single stepping of instructions where the status and PC must be preserved between consecutive nonmaskable interrupts. When NMIM=1, a nonmaskable interrupt can be used to simulate a hardware reset in software (using the NMI vector). Saving the context may be of no benefit if either:

  - Control is never to be returned to the interrupt program

  or

  - The integrity of the stack pointer is suspect.

  The nonmaskable interrupt does not cause the I/O registers to be reset. Consequently, if an NMI is used to simulate a hardware reset, the I/O registers should be reset by software within the NMI service routine.

NMI	NMIM	Effect
0	0	No effect
0	1	Undefined
1	0	NMI (save context on stack)
1	1	NMI (discard previous context)

- **CF** (*Cache flush*)

  While CF is set to 1, the contents of the instruction cache are flushed. All four P (present) flags in the cache control logic remain forced to 0 as long as CF remains 1. When CF=1, the cache is disabled; instruction words are fetched from local memory one at a time as they are needed for execution by the GSP. Normal cache operation resumes when CF is set to 0, assuming the CD bit in the CONTROL register is also 0. When the value of CF is changed from 1 to 0, the cache begins operation in the same initial state as that which immediately follows reset.

  One use of the CF bit is during downloads of new software from the host processor to GSP local memory. By setting CF to 1 and then to 0 again, the host processor forces the GSP to begin to load new instructions into the cache from memory rather than continue execution of stale instructions already contained in the cache. A 0 must be loaded into CF for normal cache operation to resume.

CF	Effect
0	No effect
1	Flush and disable cache

- LBL (*Lower byte last*)

  The LBL bit specifies whether an indirect access of GSP memory, initiated by a host register access, begins when the upper or lower byte of the register is accessed by the host processor.

  LBL is provided to accommodate host processors with 8-bit data paths. An 8-bit processor must access a 16-bit GSP host interface register as a series of two 8-bit bytes. Processors which access the lower byte (bits 0–7) first and the upper byte (bits 8–15) second should typically set LBL to 0, and those that access bytes in the opposite sequence should set LBL to 1.

  When LBL is 0, a local bus cycle is initiated if

  – The host writes to the upper byte of HSTADRH,

  or

  – The host reads from or writes to the upper byte of HSTDATA

  If LBL is 1, a local bus cycle is initiated if

  – The host accesses the lower byte of HSTDATA

  or

  – The host writes to the lower byte of HSTADRL

  With this capability, the GSP chip is capable of automatically resolving so called "Little-Endian/Big-Endian" byte addressing incompatibilities between various processors, and promotes software transparency between 8- and 16-bit versions of the same processor architecture (such as the 8088 and 8086).

LBL	Effect
0	Initiate 16-bit local bus cycle on host access of upper byte of HSTDATA, or on load of upper byte of HSTADRH
1	Initiate 16-bit local bus cycle on host access of lower byte of HSTDATA, or on load of lower byte of HSTADRL

- **INCR** (*Increment address before local read*)

  The INCR bit controls whether or not the 32-bit address pointer contained in the HSTADRL and HSTADRH registers is incremented before each read.

INCR	Effect
0	Do not increment address pointer before read cycle on local memory bus.
1	Increment address pointer before read cycle on local memory bus.

When INCR=1, the 32-bit address contained in registers HSTADRL and HSTADRH is incremented by 16 before being used for the next read of the GSP memory. This means that HSTDATA is updated to the contents of the next sequential word in the local memory in preparation for the next anticipated read of HSTDATA by the host processor. A local read cycle also occurs when the host loads a new address into the HSTADRL and HSTADRH registers, but the address is not incremented in this case. When incrementing is enabled, repeated reads of the HSTDATA register by the host result in a series of adjacent words in GSP memory being read; otherwise, the same memory word is read each time. Regardless of the value of the INCR bit, each time HSTDATA is read by the host, a new word is automatically read into HSTDATA from the GSP's memory.

- **INCW** (*Increment address after local write*)

The INCW bit controls whether or not the 32-bit address pointer contained in the HSTADRL and HSTADRH registers is incremented after each write.

INCW	Effect
0	Do not increment address pointer after write cycle on local memory bus.
1	Increment address pointer after write cycle on local memory bus.

When INCW=1, the 32-bit address contained in registers HSTADRL and HSTADRH is incremented by 16 after being used as the memory write address. When incrementing is enabled, repeated writes to the HSTDATA register by the host cause a series of adjacent words in GSP memory to be modified; otherwise, the same memory word is modified repeatedly. Regardless of the value of the INCW bit, each time HSTDATA is written to by the host, a new cycle is initiated to write the contents of HSTDATA to the GSP's memory.

- **HLT** (*Halt GSP program execution*)

When the HLT bit is set to 1, the GSP suspends instruction processing at the next instruction boundary. Once halted, the GSP does **not** respond to interrupt requests (including NMI). Local memory refresh and video timing functions continue unaffected while the GSP is halted. When HLT is again set to 0, the GSP continues execution.

The state of the HLT bit immediately following reset is determined by the state of the $\overline{\text{HCS}}$ pin at the time of the low-to-high transition of $\overline{\text{RESET}}$. If $\overline{\text{HCS}}$ is low, HLT is set to 0, and the GSP is enabled to begin executing its reset routine. If $\overline{\text{HCS}}$ is high, the HLT bit is set to 1, and the GSP is halted. Both the host processor and GSP can write to the HLT bit; this means the GSP can halt itself by loading a 1 into HLT.

HLT	Effect
0	Allow GSP to run
1	Halt GSP instruction execution

HSTCTLL Host Interface Control Register, Low Byte HSTCTLL

Address >C000 00F0

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								INT Out	MSGOUT			INT In	MSGIN		

Fields

Bits	Name	Function
0–2	MSGIN	Input message buffer
3	INTIN	Input interrupt bit
4–6	MSGOUT	Output message buffer
7	INTOUT	Output interrupt bit
8–15	Reserved	Not used

Description The HSTCTLL register contains eight programmable bits used to control host interface communications. A host processor can access the control bits in the HSTCTLL and HSTCTLH registers as a single host interface register, HSTCTL. The bits of the host interface's HSTCTL register are mapped into two separate I/O register locations in the GSP's memory map, HSTCTLL and HSTCTLH, to allow the GSP to alter the bits in one location without affecting the bits in the other.

The HSTCTLH bits can be read by both the host processor and the GSP. The following restrictions apply to writes:

- The MSGOUT field can be modified only by the GSP.
- The MSGIN field can be modified only by the host.
- The host can write a 1 to the INTIN bit, but writing a 0 has no effect.
- The GSP can write a 0 to the INTIN bit, but writing a 1 has no effect.
- The GSP can write a 1 to the INTOUT bit, but writing a 0 has no effect.
- The host can write a 0 to the INTOUT bit, but writing a 1 has no effect.

Internal arbitration logic permits the GSP and host processor to access HSTCTLL at the same time without hazard. Synchronization of asynchronous signals at the host interface pins is performed internally.

- **MSGIN** (*Message in, host to GSP*)

The MSGIN field buffers a 3-bit interrupt message to the GSP from the host. The MSGIN field can be both written to and read by the host, but only read by the GSP. The MSGIN field typically contains a command or status code from the host, which is read by the GSP in response to a host-generated interrupt (INTIN=1). The meaning of this code is defined in the software of the host and GSP.

- **INTIN** (*Interrupt in, host to GSP*)

  The INTIN bit controls the interrupt request to the GSP from the host. To generate an interrupt request, the host processor loads a 1 to INTIN. The GSP deactivates the request by loading a 0 to INTIN. An attempt by the host to load a 0 to INTIN has no effect. Similarly, an attempt by the GSP to load a 1 to INTIN has no effect. A read-only copy of the INTIN bit is available as the HIP bit in the INTPEND register. The HIP bit faithfully represents the state of the INTIN bit at all times.

INTIN	Effect
0	No interrupt request to GSP
1	Send interrupt request to GSP

- **MSGOUT** (*Message out, GSP to host*)

  The MSGOUT field buffers a 3-bit interrupt message to the host from the GSP. The MSGOUT field can be both written to and read by the GSP, but only read by the host. The MSGOUT field permits an interrupt request generated by means of the INTOUT bit to be qualified by an additional command or status code, the meaning of which is defined in the software of the host and GSP.

- **INTOUT** (*Interrupt out, GSP to host*)

  The INTOUT bit controls the interrupt request to the host processor from the GSP. An interrupt request is transmitted to the host by means of an active-low level on the $\overline{\text{HINT}}$ pin. When INTOUT is 1, $\overline{\text{HINT}}$ is driven active low; when INTOUT is 0, $\overline{\text{HINT}}$ is driven inactive high. The GSP activates the interrupt request by loading a 1 to INTOUT, and the host deactivates the interrupt request by loading a 0 to INTOUT. An attempt by the GSP to load a 0 to INTOUT has no effect. Similarly, an attempt by the host to load a 1 to INTOUT has no effect.

INTOUT	Effect
0	No interrupt request to host
1	Send interrupt request to host

Address >C000 00C0

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HSTDATA															

Description The HSTDATA register buffers data transferred through the host interface between GSP local memory and a host processor. HSTDATA can be accessed by the GSP at address >C000 00C0. It is one of the four 16-bit registers that can be accessed by the host register through the TMS34010 host interface. HSTDATA is typically accessed by the host rather than the GSP. Using the HSTDATA register, the host can either read the GSP's memory or write to it. The host initiates the indirect access through the host interface using the 32-bit pointer address in the HSTADRL and HSTADRH registers. During each indirect access, a 16-bit word is transferred between the HSTDATA register and GSP memory. The host processor can access the contents of the HSTDATA register in one 16-bit data transfer or two 8-bit transfers. When the TMS34010's on-chip processor reads from or writes to HSTDATA, **no** automatic read or write cycle takes place between HSTDATA and the memory word pointed to by HSTADRL and HSTADRH.

Address >C000 0030

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HTOTAL															

Description

The HTOTAL register is used during generation of the horizontal sync signal output to the video monitor from the GSP. It determines the duration of each horizontal scan line on the screen in terms of the number of VCLK (video clock) periods. The contents of HTOTAL are compared with the horizontal count in HCOUNT to determine the point at which the horizontal sync pulse begins, which also represents the beginning of a new scan line. HCOUNT counts from 0 to the value contained in HTOTAL. When HCOUNT = HTOTAL, the $\overline{\text{HSYNC}}$ output is driven active low on the next falling edge of the VCLK signal, and HCOUNT is reset to 0 on the same clock edge.

HTOTAL is loaded with a 16-bit value greater than that contained in HSBLNK, but less than or equal to 65535. In interlaced scan mode, the value in HTOTAL should be an odd number (LSB=1) to achieve equal spacing between adjacent scan lines. The total number of VCLK video clocks in each horizontal scan line is calculated as HTOTAL + 1. When external sync mode is enabled (DXV=0) and $\overline{\text{HSYNC}}$ is configured as an input (HSD=0), HTOTAL should be loaded with a value greater than the value of HCOUNT at the point at which the external sync pulse is expected. If the external sync pulse does not occur, HCOUNT will be reset when HCOUNT = HTOTAL.

INTENB Interrupt Enable Register INTENB

Address >C000 0110

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				WVE	DIE	HIE	Reserved						X2E	X1E	Res.

Fields

Bits	Name	Function
0	Reserved	Not used
1	X1E	External interrupt 1 enable
2	X2E	External interrupt 2 enable
3–8	Reserved	Not used
9	HIE	Host interrupt enable
10	DIE	Display interrupt enable
11	WVE	Window-violation interrupt enable
12–15	Reserved	Not used

Description The INTENB register contains the interrupt mask used to selectively enable the three internally and two externally generated interrupt requests. The following interrupts are enabled by the INTENB register:

- External interrupts 1 and 2 are generated by active-low signals on the input pins $\overline{\text{LINT1}}$ and $\overline{\text{LINT2}}$, respectively.
- The host interrupt is generated when the host processor sets the INTIN bit in the HSTCTL register to 1.
- The display interrupt is generated when the vertical count in the VCOUNT register reaches the value contained in the DPYINT register.
- The window-violation interrupt is caused by an attempt to write a pixel to a region of the bit map lying outside the limits of the currently-defined window.

The status register contains a global interrupt enable bit, IE. The INTENB register contains individual interrupt enable bits associated with each of the interrupts (X1E, X2E, HIE, DIE, and WVE). Interrupts are enabled through a combination of setting the IE bit and the appropriate bit in the INTENB register. When IE=0, all interrupts are disabled regardless of the values of the bits in the INTENB register. When IE=1, each interrupt is enabled or disabled according to the corresponding enable bit in the INTENB register (1 enables the interrupt, 0 disables it).

Address >C000 0120

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				WVP	DIP	HIP	Reserved						X2P	X1P	Rsv

Fields

Bits	Name	Function
0	Reserved	Not used
1	X1P	External interrupt 1 pending
2	X2P	External interrupt 2 pending
3–8	Reserved	Not used
9	HIP	Host interrupt pending
10	DIP	Display interrupt pending
11	WVP	Window-violation interrupt pending
15–12	Reserved	Not used

Description The INTPEND register indicates which interrupt requests are currently pending. INTPEND's six active bits indicate the status of the following interrupts:

- External interrupts 1 and 2 are generated by active-low signals on the input pins $\overline{\text{LINT1}}$ and $\overline{\text{LINT2}}$, respectively.
- The host interrupt request is generated when the host processor sets the INTIN bit in the HSTCTL register to 1.
- The display interrupt request is generated when the vertical count in the VCOUNT register reaches the value contained in the DPYINT register.
- The window-violation interrupt request is caused by an attempt to write a pixel to a region of the bit map lying inside or outside the limits of the currently-defined window, depending on the selected windowing mode.

The individual pending bits in the INTPEND register reflect the status of interrupt requests. The interrupt is requested if the corresponding pending bit is 1. There is no request if the pending bit is 0. The status of each interrupt request is reflected in the INTPEND register regardless of whether the interrupt is enabled or not; this allows the GSP to poll interrupts.

The X1E and X2E bits of INTPEND are read only. They reflect the input levels on the $\overline{\text{LINT1}}$ and $\overline{\text{LINT2}}$ pins, and are not affected when the INTPEND register is written to. If an external interrupt is disabled, the interrupt request is ignored, even though the corresponding pending flag in INTPEND is set. The interrupt will be taken by the GSP only if the external request is maintained at the corresponding interrupt request pin until the interrupt is again enabled.

The DIP and WVP bits in the INTPEND register reflect the status of interrupt requests generated by conditions internal to the GSP. These two bits are implemented as latches. Once set, DIP or WVP will remain set until a 0 is written to it (or the GSP is reset). Writing a 1 to either of these bits has no effect at any time. While an internal interrupt is disabled, the interrupt request is ignored, even though the corresponding pending flag in INTPEND is set. If the interrupt is subsequently enabled while the interrupt pending flag remains set (because of a prior interrupt request) then the interrupt will be taken by the GSP.

The HIP bit in the INTPEND register is a read-only bit that always displays the current contents of the INTIN bit in the HSTCTL register. Writing to the INTPEND register has no effect on the HIP bit. A host interrupt request is generated when the host processor writes a 1 to the INTIN bit of the HSTCTL register. The GSP clears the interrupt request by writing a 0 to the INTIN bit.

Address >C000 0160

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PMASK															

Description The PMASK register selectively enables or disables various planes in the bit map of a display system in which each pixel is represented by multiple bits. PMASK contains a 16-bit value that determines which bits of each pixel can be modified during execution of a DRAV, PIXT, FILL, LINE, or PIXBLT instruction. Via the PMASK register, the programmer specifies which bits within each pixel are protected (mask bit=1) and not protected (mask bit=0) from modification. During a pixel write operation, the 0s in the plane mask represent bit positions within the destination pixel that are to be modified by the pixel operation. The 1s in the plane mask represent bit positions in the destination pixel that are protected from modification.

The organization of a display memory is sometimes described in terms of bit planes. If the pixel size is four bits, for example, and the bits in each pixel are numbered from 0 to 3, the display memory is said to be composed of four bit planes, numbered from 0 to 3. Plane 0 contains all the bits numbered 0 from all the pixels, plane 1 contains all the bits numbered 1 from all the pixels, and so on. A 4-bit mask is constructed such that bit 0 of the mask enables (if 0) or disables (if 1) writes to the bits in plane 0, mask bit 1 enables or disables writes to plane 1, and so on.

The plane mask for a 4-bit pixel is four bits; the plane mask for an 8-bit pixel is eight bits; and so on. The plane mask must be replicated throughout the 16 bits of the PMASK register. For example, with four bits per pixel, the PMASK register is loaded with four identical copies of the corresponding 4-bit plane mask, as indicated below.

	15 — 12	11 — 8	7 — 4	3 — 0
PMASK	MASK	MASK	MASK	MASK

With a pixel size of eight bits, the corresponding 8-bit plane mask is replicated twice – once in bits 0–7 of PMASK, and again in bits 8–15. In general, all 16 bits of the register are used, and a mask for a pixel size of less than 16 bits must be duplicated n times, where n is 16 divided by the pixel size.

The individual bits of the PMASK register are associated with the corresponding bits of the 16-bit local data bus (data are in fact multiplexed over the same LAD0–LAD15 pins as addresses). PMASK register bit 0 is associated with bit 0 of the data bus (the bit transferred on LAD0), PMASK bit 1 is associated with bit 1 of the data bus, and so on. In general, if PMASK bit n is a 0, then bit n of the data bus is enabled by the mask; if PMASK bit n is a 1, bit n is disabled by the mask.

Plane masking is effectively disabled (allowing all bits of each pixel to be modified) by loading all 0s into the PMASK register. This is the default state of PMASK following reset.

To maintain upward compatibility with future versions of the GSP, software drivers should tread the PMASK register as a 32-bit register beginning at address >C000 0160. In other words, software should write the plane mask value not only to the 16-bit word at address >C000 0160, but also to the word at >C000 0170. Writing the second word will have no effect on the TMS34010, but will ensure software compatibility with future graphics processors which may extend the PMASK register from 16 to 32 bits.

Address >C000 0150

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSIZE															

Description The PSIZE register is used to specify the pixel size in bits. If the pixel size is four, for example, PSIZE is loaded with the value four. If the pixel size is eight, PSIZE is loaded with the value eight, and so on. All 16 bits of the PSIZE register can be written to or read. Legal pixel sizes are 1, 2, 4, 8, and 16 bits; any other value of PSIZE is undefined.

PSIZE	Pixel Size
>0001	1 bit/pixel
>0002	2 bits/pixel
>0004	4 bits/pixel
>0008	8 bits/pixel
>0010	16 bits/pixel

Address >C000 01F0

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ROWADR								RINTVL						Reserved	

Fields

Bits	Name	Function
0–1	Reserved	Not used
2–7	RINTVL	Refresh interval
8–15	ROWADR	Row address

Description

The REFCNT register generates the addresses output during DRAM refresh cycles and counts the intervals between successive DRAM refresh cycles.

DRAMs require periodic refreshing to retain their data. The GSP automatically generates DRAM refresh cycles at regular intervals. The interval between refresh cycles is programmable. The DRAM refresh mode is selected by loading the appropriate value to the two-bit RR (refresh rate) field in the CONTROL register. DRAM refreshing can be disabled in systems that do not require it. The modes are defined as follows.

RR	Description
00	Refresh every 32 local clock periods
01	Refresh every 64 local clock periods
10	Reserved for future expansion
11	No DRAM refreshing

At reset, the RR field is set to the initial value 00. During the time that the reset signal to the GSP is active, no DRAM-refresh cycles are performed.

Bits 2–15 of REFCNT form a continuous binary counter. Bits 2–7 form the RINTVL field, which counts the intervals between successive requests for DRAM-refresh cycles. When RR=01, the RINTVL field is incremented by 1 every local clock cycle; that is, the register is incremented at bit 2. This means that RINTVL overflows into ROWADR (a carry ripples from bit 7 to bit 8 of REFCNT) every 64 local clock cycles. The overflow has two effects:

- ROWADR is incremented by 1.
- A request for a DRAM-refresh cycle is sent to the memory control logic.

When RR=00, the RINTVL field is incremented by 2 every local clock period. This means that a DRAM-refresh cycle is generated every 32 local clock periods, twice the rate that results when RR=01. When RR=11, DRAM refreshing is disabled and no DRAM-refresh cycles occur.

During a DRAM-refresh cycle, the row address output to memory is taken from the 8-bit ROWADR field of REFCNT. Specifically, bits 8–15 of REFCNT are output on LAD0–LAD7. REFCNT bits 8–14 are simultaneously output on LAD8–LAD14. (The $\overline{RF}$ bus status signal is output as a low level on LAD15.) This means that the 8-bit row address needed to refresh a DRAM can be taken from any eight adjacent LAD pins in the range LAD0–LAD14. Note that as ROWADR counts from 0 to 255, the refresh addresses output at the selected eight LAD pins will sequence through all 256 values in the range 0 to 255, though not necessarily in the same order as ROWADR.

Address >C000 01D0

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VCOUNT															

Description

The VCOUNT register is a 16-bit counter used during generation of the vertical sync and blanking signals. VCOUNT counts the horizontal lines in the video display, incrementing at the same clock edge at which HCOUNT is internally reset to 0. This causes the falling edges of $\overline{\text{HSYNC}}$ and $\overline{\text{VSYNC}}$ to coincide.

In order to generate vertical sync and blanking signals, the value of VCOUNT is compared to the value of the four vertical timing registers, VESYNC, VEBLNK, VSBLNK, and VTOTAL. When HCOUNT = HTOTAL and VCOUNT = VTOTAL at the same time, VCOUNT is reset to 0 on the next VCLK falling edge and the $\overline{\text{VSYNC}}$ output is driven active low.

If interlaced scan mode is enabled and the current field is *even*, and if VCOUNT = VTOTAL and HCOUNT = HTOTAL/2, then VCOUNT is reset to 0 and $\overline{\text{VSYNC}}$ goes low (HCOUNT is not reset until it reaches the value HCOUNT = HTOTAL). When external sync mode is enabled, VCOUNT is reset to 0 when the $\overline{\text{VSYNC}}$ input signal goes active low.

A display interrupt request is generated when VCOUNT = DPYINT. This can be used to coordinate software activity with the refreshing of selected lines on the screen.

Two separate, asynchronous elements of the GSP internal logic can access VCOUNT:

- The internal processor, which runs synchronously to local clocks LCLK1 and LCLK2, can access VCOUNT as an I/O register.
- The video timing control logic, which runs synchronously to the video clock VCLK, increments and clears VCOUNT in the course of generating the sync and blanking signals.

No synchronization between these two subsystems is provided, and VCOUNT can only be reliably read or written while VCLK is held at the logic-high level. VCOUNT is typically not read or written to except during chip test.

Address >C000 0050

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VEBLNK															

Description VEBLNK is a video timing register that designates the time at which the vertical blanking interval ends. The 16-bit value contained in VEBLNK is compared to VCOUNT to determine when to end the vertical blanking interval. The vertical blanking interval ends when the following conditions are satisfied:

- VCOUNT = VEBLNK
- HCOUNT = HTOTAL

The end of the vertical blanking interval coincides with the start of the horizontal sync, occurring at a time when the internal horizontal blanking signal is active. The blanking signal output from the $\overline{\text{BLANK}}$ pin is a composite of the horizontal and vertical blanking signals generated internally, and will not reach its inactive-high level until both internal blanking signals have become inactive.

When external video is enabled (DXV=0) and the $\overline{\text{HSYNC}}$ pin is configured as an input (HSD=0), the vertical blanking interval ends when the following conditions are satisfied:

- VCOUNT = VEBLNK
- The leading edge of the external horizontal sync pulse is detected

The beginning of the sync pulse is seen as a high-to-low transition at the $\overline{\text{HSYNC}}$ pin.

Typical video monitors require VEBLNK to be set to a value less than the value in VSBLNK, and greater than the value in VESYNC.

Address >C000 0040

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VESYNC															

Description

VESYNC is a video timing register that designates the time at which the vertical sync pulse ends. The 16-bit value contained in VESYNC is compared to VCOUNT to determine when to end the vertical sync pulse. The sync pulse ends when the following conditions are satisfied:

- VCOUNT = VESYNC
- HCOUNT = HTOTAL

The $\overline{\text{VSYNC}}$ output is driven inactive high to signal the end of the vertical sync interval.

When interlaced mode is enabled and the next vertical field is odd, $\overline{\text{VSYNC}}$ is driven high when VCOUNT = VESYNC and HCOUNT = HTOTAL/2.

Typical video monitors require VESYNC to be set to a value less than the value contained in the VEBLNK register; the minimum value of VESYNC is 0.

When external sync mode is enabled (DXV=0), the end of the external vertical sync pulse is detected as a low-to-high transition at the $\overline{\text{VSYNC}}$ pin, which is configured as an input. VESYNC should be loaded with a value greater than the value in VCOUNT at the point at which the external $\overline{\text{VSYNC}}$ input signal should go inactive high, but lower than the value in VCOUNT when the external $\overline{\text{VSYNC}}$ should again become active low. For example, VESYNC could be loaded with the sum of the values in VEBLNK and VSBLNK divided by two.

VSBLNK — Vertical Start Blank Register — VSBLNK

Address >C000 0060

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VSBLNK															

Description

VSBLNK is a video timing register that designates the time at which the vertical blanking interval starts. The 16-bit value contained in VSBLNK is compared to VCOUNT to determine when to start the vertical blanking interval. The vertical blanking interval starts when the following conditions are satisfied:

- VCOUNT = VSBLNK
- HCOUNT = HTOTAL

The start of the vertical blanking interval coincides with the start of the horizontal sync, occurring at a time when the internal horizontal blanking signal is active. The blanking signal output from the $\overline{\text{BLANK}}$ pin is a composite of the horizontal and vertical blanking signals generated internally, and reaches its active-low level when either or both internal blanking signals are active.

When external video is enabled (DXV=0) and the $\overline{\text{HSYNC}}$ pin is configured as an input (HSD=0), the vertical blanking interval starts when the following conditions are satisfied:

- VCOUNT = VSBLNK
- The leading edge of the external horizontal sync pulse is detected

The beginning of the horizontal sync pulse is seen as a high-to-low transition at the $\overline{\text{HSYNC}}$ pin.

VSBLNK should be set to a value less than the value in VTOTAL, and greater than the value in VEBLNK.

Address >C000 0070

Bit Assignments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VTOTAL															

Description

VTOTAL contains a 16-bit value that designates the value of VCOUNT at which the vertical sync pulse begins. The contents of VTOTAL are compared to VCOUNT to determine when to start the vertical sync pulse. Vertical sync begins when the following two conditions are satisfied:

- VCOUNT = VTOTAL
- HCOUNT = HTOTAL

These conditions cause HCOUNT to begin counting from 0 again.

The $\overline{\text{VSYNC}}$ output is driven active low to signal the start of the vertical sync interval. The high-to-low transitions of $\overline{\text{VSYNC}}$ and $\overline{\text{HSYNC}}$ occur at the same clock edge.

When interlaced mode is enabled and the next vertical field is odd, $\overline{\text{VSYNC}}$ is driven low when VCOUNT = VESYNC and HCOUNT = HTOTAL/2. The total number of horizontal lines in each vertical field is calculated as VTOTAL + 1. In interlaced mode the total number of horizontal lines in both fields of the vertical frame is calculated as 2×VTOTAL-1.

When external video is enabled (DXV=0), the $\overline{\text{VSYNC}}$ pin is configured as an input rather than an output. The high-to-low transition of $\overline{\text{VSYNC}}$ is recognized as the beginning of the vertical sync pulse, unless the condition VCOUNT = VTOTAL and the start of horizontal sync are detected first. VTOTAL should be loaded with a value at least as large as the value of VCOUNT at which the external sync pulse should begin. Should the external sync pulse not occur, VCOUNT will be reset one VCLK period after the conditions VCOUNT = VTOTAL and HCOUNT = HTOTAL occur.

VTOTAL should be set to a value greater than the value in VSBLNK. The maximum value that can be loaded into VTOTAL is 65535.